VTT INFORMATION TECHNOLOGY

RESEARCH REPORT TTE4-2002-40

MobTv

A method to deliver multiple media content for digital television

Version 1.0

29.11.2002

Mikko Kojo

Version history

Version	Date	Author(s)	Reviewer	Description
1.0	29.11.2002	Mikko Kojo		Final version

Contact information

Mikko Kojo
VTT Information Technology
P.O. Box 12041, FIN-02044 VTT, Finland
Street Address: Tekniikantie 4 B, Espoo
Tel. +358 9 4561, fax +358 9 456 7052
Email: Mikko.Kojo@vtt.fi
Web: http://www.vtt.fi/tte/

Last modified on 29 November, 2002
H:\Eudora\liitteet\A method to deliver multiple media content for digital television.doc

Abstract

In multiple media the same information is delivered in more than one medium. Newspapers and the Internet are examples of such media. Digital television broadcasts have begun in Finland. More interactive services are possible in digital television, and information published in other media can be delivered to television more easily. In this report, multiple media and digital television are introduced. Methods to use digital television as a medium in multiple media are presented. A prototype application is developed to test one method in practice, and the process of developing the prototype is reported. The project itself and digital television's usability in multiple media are evaluated.

Contents

List of symbols

API	Application programming interface
ASF	Advanced streaming format
AVI	Audio video interleaved
CGI	Common gateway interface
CORBA	Common object request broker architecture
CPU	Central processing unit
CSS	Cascading stylesheets
DASE	Digital TV applications software environment
DAVIC	Digital audio video council
DVB	Digital video broadcasting
EB	Enhanced broadcasting
ECMA	European computer manufacturers association
EPG	Electronic program guide
FPS	Frames per second
GUI	Graphical user interface
HAVi	Home audio video interoperability
HTML	Hypertext markup language
HTTP	Hypertext transfer protocol
IDE	Integrated development environment
I/O	Input/output
IRT	Institut für Rundfunktechnik
IB	Interactive broadcasting
J2EE	Java 2 platform enterprise edition
JMF	Java media framework
LDU	Logical data unit
MHP	Multimedia home platform
MP3	MPEG-1 audio layer 3
NIST	National institute of standards and technology
MPEG	Motion picture experts group
MVC	Model view controller

PC	Personal computer
PDA	Personal digital assistant
PHP	PHP: hypertext preprocessor
RMI	Remote method invocation
SDK	Software development kit
SMS	Short message service
TCP/IP	Transmission control protocol / Internet protocol
UI	User interface
UML	Unified modeling language
VTT	Technical research centre of Finland
WAP	Wireless application protocol
WWW	World wide web
XHTML	Extensible hypertext markup language
XML	Extensible markup language

1 Introduction and objectives

This report is slightly modified version of Mikko Kojo's master's thesis. The thesis was done for University of Helsinki's Department of Computer Scienceat VTT. It is part of the project *"Mobile television in fourth generation networks"* [Mob01]. The acronym MobTV is used for the project in this report.

In multiple media the same information is published in more than one medium. Consumers are nowadays using various new information sources in addition to more traditional television and newspapers. They are accessing the information with various digital devices. At the same time co-operation between media, telecommunications and information technology companies is increasing. These two things either make multiple media possible or require it, depending on the standpoint.

Television is currently going through a digitalisation process, which could result in its role being changed. It may become more interactive as an information source and it could become one of the media used in multiple media. The objective of this research is to study the methods that can be applied to achieve this. One of the methods is tested in practice with a developed prototype application. The main motivation of this research for the MobTV project, and also one of the project's objectives, is to test the suitability of MHP for the needs of this project.

The first two chapters, which are based on the source material, are about the background technologies. Multiple media is discussed in chapter 2. Some basic terminology is explained first. Next, media convergence as a background force behind multiple media is discussed. Then the advantages of multiple media in a form of synergy are covered. Sections 2.4 and 2.5 are about the content: different content elements are presented first, and then content management is discussed. The characteristics of different media are gone through next. The chapter ends with a discussion about adaptation, which means tailoring of the content, and the content's appearance and arrangement.

Digital television is the topic of chapter 3. Firstly, some background, the current situation, and the outlook for the near future are presented. Next, some typical characteristics of the television platform are discussed. Section 3.2 is more technical, and about the Multimedia Home Platform, MHP, which is a specification intended to ensure that applications are compatible with digital television terminals.

Chapter 4 introduces possible methods to deliver multiple media content to MHP compliant digital television. One of the methods is tested in practice with a prototype application. The first section of chapter 5 presents the arguments as to why the prototype is needed. Section 5.2 describes the software project, in which the prototype application is developed with a focus on general issues. In section 5.3, further development of the software is covered.

The last chapter is a discussion of the research itself, MHP in practice and its suitability for multiple media. Firstly, the whole research project is evaluated. Then the experiences

in MHP application development are reported. The last section of chapter 6 is about the use of MHP in multiple media.

The MobTV project group has made a number of decisions regarding the progress of the project, and these decisions are mentioned separately in the thesis.

2 Multiple media

2.1 Terminology

In the following section the multiple media terminology used in this thesis is defined. *Multiple media* itself means that the same information, *content*, is delivered to end-users in more than one *medium*. A medium can be defined as a means of mass communication, such as newspapers, radio or television [Ame00]. In telecommunications a medium is the transmission path along which a signal propagates [FS-1037C]. Neither of the definitions clashes with the meaning that medium has in multiple media. In multiple media, and in this thesis, medium is the means by which information is delivered from the information provider to the end-user. Therefore medium is more than just a transmission path. The first definition is closer to the meaning of medium in multiple media. However, there are media that are not so clearly mass communication as, for example, a newspaper is. For example, it is possible to deliver information directly to a user's cellular phone.

When content is made available in one or more media, it is *published*. This is done by the *content provider*.

Multiple media is constantly confused with *multimedia*, which means presentation of information in more than one form [FS-1037C]. Normally, text combined with images is not considered multimedia. The content in multiple media can be, and often is, multimedia. The content and the subsystems that are needed to handle the content and to publish it in various media are called a *multiple media system*.

The most common example of multiple media is a newspaper and its WWW site. The content or some of the content that is printed in the newspaper is also available on the publisher's WWW site. In this case there are two media: the newspaper medium and the WWW medium. The multiple media system consists of databases, editing tools, servers, etc.

2.1.1 Additional Terminology

Multiple media terminology is not well established. In this section some additional terminology is presented. These terms are not used in the thesis, but are gone through here as they are frequently used in the literature.

While medium's plural is media, the term media has another meaning. In spoken language the media can mean the communities and institutions behind the forms of communication. The media is then used as a singular with the definite article. In this sense the media could mean, for example, something like the press. [Ame00]

Sometimes content provider means only the creator of content, the one the content originates from. The *syndicator* assembles the service using content from various content providers. The term syndicator is not used in this thesis. In the thesis the content provider has access and rights to the content and makes it available for users. It is possible that the content provider is not the original source of the content.

The term multiple media is often mistakenly used as a synonym of multimedia. Alternative terms also exist. One of the most common is *multichannel publishing*, which means the same thing as multiple media. The same content is published on various *channels*, i.e. media. The terms should not be used in conjunction. The main concept of multiple media is sometimes described by the term *cross-media publishing* [Neu00].

Medium is used in a completely different sense when it means an easily recognisable form of information product or service. In connection with medium in this sense, the term *channel* is used. It has no clear equivalent in the terminology defined in 2.1. If the previous section's example were to be described with these terms, the newspaper medium is distributed in two editions – the newspaper edition through the newspaper channel and the WWW edition through the WWW channel. [SNE97]

2.2 Convergence

Convergence in general means the act of coming closer or moving toward uniformity. In this context it means that business sectors, and also the production and the delivery technologies, are becoming more alike. In multiple media, those business sectors are media, telecommunications, and information technology [HeH00] [Neu00]. Also the term *media convergence* is used, when different media have converged.

Media convergence makes it easier, or even necessary, for a content provider to publish the content in various media. Convergence is often a result of the merging or integration of companies operating in different fields that overlap each other. These fields used to be separate, but they are now converging or even merging. It should be noted that convergence is not integration, and it does not require co-operation. [HeH00] [Neu00]

There are three factors that initiate convergence: technological innovations, co-operation between companies, and changes in consumer behaviour [Sti99]. The most important and fundamental technological innovation behind convergence is the storing of information in a digital form. Other important technological innovations include the Internet (especially WWW), mobile devices and digital television.

Telecommunications and information technology have long been associated with companies like Nokia and Sonera. Their convergence with media companies is not so common. Many media companies have launched their WWW services, so some overlap with information technology can be found. However, a good example of convergence is AOL Time Warner, the merger of media company Time Warner Inc. and Internet service

provider America Online Inc. In Finland an example is the merger of media company Hämeen Sanomat Oy and telecommunications company Hämeen Puhelin Oy.

Consumers have begun to use the Internet as a source of information in addition to traditional media (newspapers, radio, television). This is one of the changes in consumer behaviour that leads to convergence. The Internet and its content do not have any relevance if consumers do not use it to access the content.

Convergence affects the way in which digital content is distributed. All traditional content will not be folded into a single distribution channel; rather, entirely new media will emerge. Also the content itself changes. New devices to access information make possible new ways to use and combine multimedia content [HeH00].

Broadcasters, publishers and Internet content providers that begin to operate on converged markets try to exploit as many media as possible. New companies also enter the converged markets [Neu00]. For these companies it is important to be able to create a single information product using various media. Consumers are accessing information through a growing variety of media and therefore multiple media enhances the brand visibility. It can also improve quality and cut costs because the content can be re-used, re-purposed and redistributed [HeH00]. Building an automated multiple media system still involves more than recycling content used elsewhere [Söd01].

Consumer behaviour can be tracked in some digital media. This has not really been possible in traditional media like newspapers and analogue television. Although a publisher knows who subscribes to a newspaper, it is impossible to know whether the subscriber reads anything else but the comics. In digital media it is easier to measure the usage. It is possible that a user has to request every piece of content he wishes to receive. To enable that, the user and his requests have to be identified. This makes it possible to track user actions. In traditional media the whole content is delivered to the user without any direct request. For example, a user may subscribe to a newspaper that he seldom reads, or he may leave the television on while he is not present.

Even if the user actions can be tracked, it could still be hard to calculate the effectiveness or profitability of a given piece of media [HeH00]. The tracking of user behaviour leads to several benefits anyhow. The user identification makes it possible to create user profiles. Profiles in turn make one-to-one marketing, personalised advertising and adaptation of the content possible. User profiles are discussed in section 2.7. [Neu00] [SNE97]

2.3 Synergy in multiple media

The benefits that a company derives from multiple media are mainly due to synergy. There is synergy on four levels in multiple media [SNE97].

1) Content synergy means that parts of or all the content from one medium are used in another medium. One extreme in producing content for several media is to keep the processes separate and to publish different content in all the media, in which case there is no synergy between the media. The other extreme is to convert the content in one medium into a suitable file format for publishing it in another medium. Copyright issues can be an

obstacle to content synergy. The consent of the copyright holder is needed to publish the same content to various media.

2) Parts of the production system are used in more than one medium.

3) Employees' skills can be used in more than one medium.

4) Brand name value and identity can be transferred between the media. The number of exposures to the customers grows, the medium can be advertised in other media, and customer service is easier to handle in some media.

2.4 Content elements

Content elements are the individual items of published information. In multiple media there can be elements of the following types [SNE97].

Text elements

Image elements

Video elements (including animations)

Audio elements

The latter two elements differ from the first two. Text and images are static. Audio and video elements are dynamic by nature and their appearance changes. The change in appearance is tied to time. A more in-depth discussion of audio and video elements is presented in section 2.4.1.

There is a fifth element type, a compound element. It consists of more than one element. Compound elements are typically delivered to the client instead of more primitive elements: text, image, audio, or video. Normally, text is accompanied by images and video is accompanied by audio. A radio broadcast is an example where there could be only one element that is not a compound element. Sometimes it is necessary to divide an element into smaller elements to form a compound element. For example, a book can be divided into chapters and the chapters can be divided into paragraphs. The compound element 'book' contains only text, but it might still be appropriate to manage it as a group of text elements instead of a one big text element.

It should be noted that all element types are not suitable for all media. PDAs might not have audio capabilities. Delivering audio as such to them is not sensible. For example, audio could be converted to text. That also applies to the printed media. While a PDA might be capable of playing a video clip, it should be discarded from a printed medium or converted into one or more image elements presenting the original video element.

2.4.1 Audio and video fundamentals

The focus of the implementation part of this research project is more on audio and video than on text or images. They are also less well-known to most people. The fundamentals of audio and video are presented in this section.

Both the audio and the video data are normally considered as streams. Both the video stream and the audio stream are usually embedded in the same file if they need to be presented in conjunction. These streams consist of consecutive logical data units (LDUs). In the case of audio stream, LDUs are individual samples or block of samples. With video, LDU is a frame [Ste96]. Frame rate is one property of a video clip. It is normally calculated in frames per second (fps). It defines how many frames the player should produce from the video stream in one second.

When video or audio is played, the process consists of several phases. Firstly, a file is read from the storage and passed to a player. The player has to understand the file format. AVI is an example of an audio and video file format. In the file, the video and the audio stream are normally in a compressed form. LDUs are decompressed and then played (in the case of audio) or drawn to screen (in the case of video). The term *codec* refers to a software component that implements a video or audio compression algorithm. It is an acronym for compressor/decompressor, but it is often used even if it is only capable of decompression.

There are several compressions algorithms that can be used. A player, which is a component that presents the audio and the video, has to 'understand' the compression algorithm used. That is to say, the player has to have an appropriate codec.

There are two ways to implement the codec: with software only or with both software and hardware. A codec can utilise hardware only if the computer has the specialised hardware installed. There has to be some hardware-specific software in addition to hardware itself, i.e. at least the drivers. Hardware utilisation would be the preferred choice because decompression with specialised hardware is far more effective than software decompression. If hardware for decompression is not available, the codec has to be implemented entirely with software. This has also its advantages. The codec is more universal because it does not require any specialised hardware. The setback of software decompression is that it requires lots of computation. Software compression can also consume more energy, which can be an issue with portable devices. In this respect, the used compression algorithm has a major effect. Some algorithms are lighter to decompress. Used compression also affects the quality and the file size of the video and the audio to a great degree.

The simplest way of playing audio or video from a network is to download the whole file to the local storage, such as solid state memory or hard disc drive. It is then played locally. Video files in particular can be very large, and it will take a significant amount of time to download the file before the playback can begin. The solution to this is *streaming*. With streaming, video and audio are presented at the same time as they are downloaded from the network. Playback can start almost immediately after the downloading begins. The downloading continues in the background while the video and the audio are presented.

Sometimes, not all LDUs can be decompressed. For example, in the case of streaming there could have been errors in the network. It is also possible that there are not enough resources for a codec to decompress all LDUs. Buffers are normally used to eliminate this. LDUs are decompressed to a buffer before being played, even when streaming is not used. It is still possible that the buffer is not large enough. If a LDU is damaged, there are couple of solutions to conceal errors. One solution is to expand each LDU. In the case of video, this means that each frame is viewed for a longer period of time. If the frame rate

is 25 fps, each frame is viewed for 40ms. Now if one of the 25 frames is damaged, each frame can be viewed for 41.7ms. This method might be the most elegant one but it is hard to implement. [Ste96]

The other way is to use the previous LDU twice. For example, one frame is viewed twice [Ste96]. Human perception can tolerate some kinds of glitches. With audio, humans detect changes in an audio level, not the absolute audio level. Therefore drop-outs are easy to notice and they are distracting. If some data is lost in an audio stream, level changes should not be caused. Instead of drop-out, the previous data can be used. Although it is not same as the lost one, the effect is not so distracting. With video, quality differences are easy to notice only if comparison can be done. Humans cannot detect the frame rate. If one frame is missed, the previous frame can be used without distracting effect. [Kit97]

The third alternative is to skip the damaged LDU. This is possible only when one LDU is not available, not when there are not enough resources to decompress LDUs. Trying to keep the same temporal relationship as when the stream was captured is called *intrastream synchronisation* [Ste96]. All the techniques mentioned above can be used in intrastream synchronisation.

Videos are very often presented with audio. Video and audio streams have to be synchronised. This is called *interstream synchronisation* [Ste96]. If the streams are out of sync, the effect is very distracting. The easiest way of spotting that the audio and the video stream are out of sync is when the speech has to match the mouth movements of the speaker. This is often referred as *lip sync*. The time difference between the related audio and video LDUs is called *skew* [Ste96]. Different amount of skew is said to be acceptable and it depends on the material. Research reported in [Ste96] found that when the skew is between -80ms and +80ms, the streams can be considered to be in sync. If the skew is between -120ms and –80ms or between +80ms and +120ms, the skew is detected if the speaker was near the camera. These figures give a general idea of the accuracy requirements in interstream synchronisation. Human perception is individual and the quality of the video affects these values to a great extent.

The playing of the synchronised streams has to begin at the right position, exactly at the same time, and run at the same rate [Sul98]. Now if a LDU is skipped in the case of a damaged LDU, the rate changes. Therefore this solution is not preferred because the skew increases with every skipped LDU. Only the other two techniques presented earlier are appropriate for interstream synchronisation also.

2.5 Content management

The biggest problem in multiple media is the interoperability between the media. It is possible that one medium is completely digital while the other one is not. Even in the digital media numerous file formats are used for text, images, audio and video. Also the protocols, compression rates, and resolutions depend on the client terminal, the applications and the network, and therefore they can vary from media to media. Well thought-out content management is essential to effectively handle the problems in the interoperability between the media.

To take full advantage of synergy it is essential to share at least part of the content between different media. If an entirely separate version of the content is generated for each medium, there are no advantages from content synergy. More time and money is needed when the same content cannot be used. The content can be *re-purposed* or *re-edited* for use in more than one media. Re-purposing means that just the essential technical conversions are done. Re-purposing is easy to do automatically and it is therefore inexpensive. Re-editing means that the content is edited to suit the needs of a medium. Instead of re-editing, the term reprocessing can also be used. For example, text can be shortened or rewritten. Re-editing often requires manual work and it is therefore more expensive than re-purposing. On the other hand, it makes it possible to adapt and optimise the content better for different media. Manual re-editing leads to various versions of the content and increases the need for storage. As a side effect of the increased need for storage, content has to be stored in different locations. Several versions and storage locations make it harder to manage the content effectively. [HeH00] [SNE97]

There are two approaches to the production of content for more than one media. The first approach is to produce the content for the primary medium and then derive the content for secondary media from the primary medium. The publisher concentrates on the primary medium, the content of which is then re-edited or re-purposed for additional secondary media. [SNE97]

The other approach is to produce content for symmetrical media. The content is stored in a medium-independent form. The content is re-edited or re-purposed from that form for different media. [SNE97]

Traditional newspapers or television news cannot be seen as multiple media systems, because there is only one medium. They still follow the same principle in a sense. News agencies like Reuters and STT can be seen as content storages. They offer the same content to various media. For example, the same news originating from a news agency is published in many newspapers. This can be seen as re-purposing when the news is published as such. On the other hand, the content is clearly re-edited when news from Reuters is translated into Finnish for publication in a Finnish newspaper.

The general architecture of one multiple media system is presented in figure 1. The content is published in four media: a printed medium, a WWW medium, a television medium and one additional medium. The content elements are stored in the database. All the media share the content. In the figure none of the media is a primary medium. The content is published in various media with the publishing systems. In the figure the publishing systems are separate. They could share parts to gain advantages from the synergy. In some cases the same publishing system can be used to generate all the media.

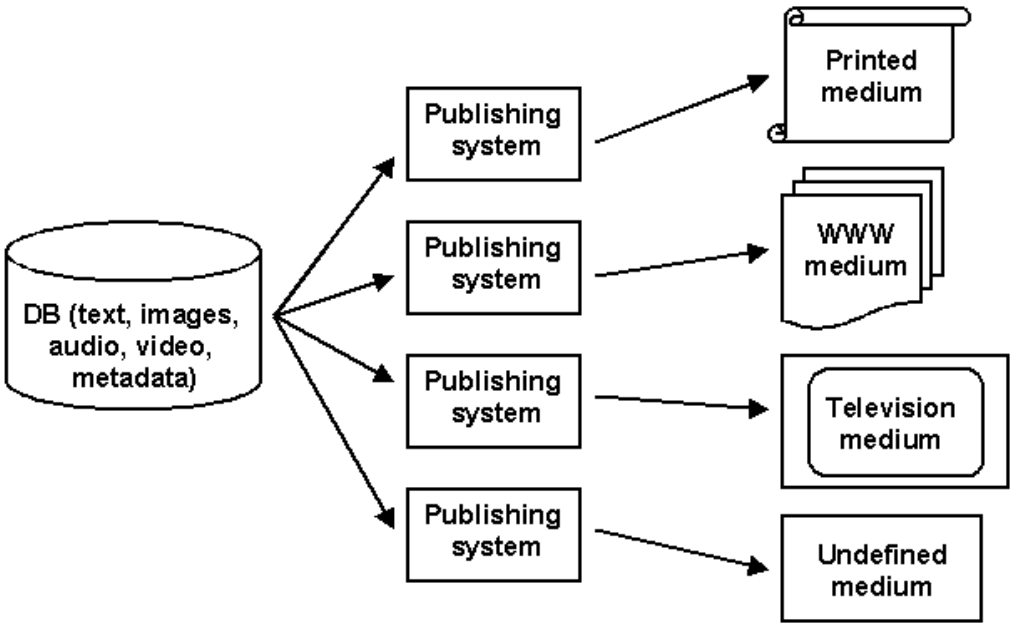

Figure 1 Example of a multiple media system.

In figure 1, in addition to the text, image, audio, and video content elements, there is *metadata* in the database. The general definition of metadata is that metadata is data about data. It answers who, what, when, where, why and how questions about the data. Metadata is the basis for the user to decide whether or not the data is appropriate for him. Metadata is almost always textual. It should also be readable by machines to automate its usage. Metadata can be divided into two categories: physical metadata like the compression rate, the size or the duration; and descriptive metadata like the author, the language or the topic [Mou01].

In multiple media metadata is needed to effectively index, search, handle, deliver, and possibly modify the content. There are no upper or lower limits on how much metadata is needed. It has to be decided each time individually. It is better to store too much metadata, because the requirements might change later. Later it could be laborious or even impossible to add the metadata. However, exaggeration should be avoided.

The content elements (text, images, ...) and *the form* (appearance and arrangement) have to be uncoupled [Agn97]. Then the content can be handled in a uniform way, despite the appearance or the format that it will have when it is published in some medium. The storing format has to be as general as possible and it should not lose valuable information, which may happen in compression. Sometimes loss of information cannot be avoided, but the format and the compression rate have to be considered carefully. For example, it is estimated that an uncompressed full-length motion picture can consume over 7.5 terabytes of storage [HeH00].

If several versions of the content for different media are stored separately, the need for content storage grows. This is most harmful with storage-consuming content elements like audio and especially video.

If the content is stored in a format optimised for some specific medium, it could be hard to use it in other media. WWW pages are sometimes edited to be viewed on devices other than PCs with web browsers. In practice, HTML does not contain any metadata; it contains only information about the desired appearance. Therefore the task is not easy. For example, even finding the document's title could be hard if only the appearance was kept in mind when the HTML document was written. This example demonstrates that the medium-independent storage format and metadata is a preferred solution for multiple media.

When the content is published in a one medium, the content and the form are coupled again. The conversion and the re-purposing of the content for the new media should be done automatically if possible. Automated systems are more expensive and they are harder to design and to implement than manual systems. Still, when the amount of content grows, the task will become too big if it is not automated. Other aspects that endorse the automated systems are frequent updates or changes, many media, and interdependent data. The automated systems require a willingness to invest in and to support the system as well as a commitment for a longer period of time. The manual systems are better suited when a response to unique situations is needed. [HeH00][Ang97]

Consumers expect that the information available is up to date. It is not enough to broadcast old news on television or deliver old newspapers to subscribers. The content has to be updated. Information in all media has to be current and kept up to date if possible. The information can become either false or outdated with time. The latter is not as critical as the former. If false or outdated information is available for the users from archives, the publishing date of the information must be clear to the user, and also the fact that the information may not be up to date. [Ang97]

Publishers are not always the copyright holders of the data they publish. It could be the case that it is forbidden to publish the data in more than one agreed medium. A copyright holder might also prevent storage of the data in the archives for customer access or allow the data to be available only for a certain period of time. [SNE97]

2.6 Characteristics of different media

Different media have different functions. Newspapers are used mainly for information, while television is for entertainment. While the reasons for the usage differ, so too do the ways in which they are used. Television can be watched quite passively, while a PC is used actively. These differences are somewhat forced by the technical characteristics of the medium. PCs have advanced input devices like a keyboard and a mouse, while a remote controller is normally used to control a television set. Mobile phones and PDAs have small screens that are not suitable for video or large blocks of text. They might also lack audio capabilities, unlike a television or a PC. Printed media differ greatly from electronic media. [Sti99]

The characteristics of the medium and its usage have to be taken into account. In multiple media different media are meant for different purposes. Characteristics are not only limitations but also potentialities that can be taken advantage of [Söd01]. There will certainly be some overlap between the media, but the media should support each other, not compete with each other [Agn97]. It is proposed that parts of the content are delivered

to the different terminals simultaneously to support each other [HPN00]. In this way the assets of the different terminals could be exploited. For example, when a PDA does not have audio capabilities, audio could be delivered to a MP3 player. This approach doesn't fit in with all situations and the synchronisation problems between the terminals have to be solved.

2.7 Adaptation

The content and its form do not have to be alike in every media. This tailoring of the content and its appearance and arrangement is called *adaptation*. Adaptation can be based on the medium. In some of the media it is possible to adapt the content and the form to suit the needs of the user.

If the content is stored in the medium-independent format, it is coupled with the form when it is published in a new medium. The content's appearance can be tailored for the medium, keeping the characteristics of the medium in mind. The content might need some re-editing or re-purposing, so that the content itself or its arrangement can change. For example, long articles are shortened for viewing on a portable device with a small screen.

The content and its form are not the only thing that can be tailored. The content should be stored as plain data elements without any functionality. In addition to form, application and user interface logics are attached when the content is published. It is possible that every media share the same logic, but sometimes this is not appropriate. The application and the user interface logic can also vary between the electronic media. At least partially different publishing systems are then needed for each such medium, but the content is still shared between the media.

Adaptation can also be based on the user. Adaptation can be automatic, in which case the term *personalization* is used. If the user controls the adaptation, it is called *customization* [Rit01].

User models are essential to user-based adaptation. User models are also referred to as user profiles. Information about the user's preferences and behaviour is gathered in the user model. This information is then analysed. Based on the analysis, the right information in the right form is delivered to the user, in some cases even at the right time.

In personalization the analysis is based on the user's previous behaviour and other users' behaviour. If two users seem to view largely the same information, some information viewed by one user can be recommended to the other. This is just a simplified example, and many simple and more complex algorithms for personalization are available. Personalization is an on-going process. It is continually done based on the changes in the user models. [Rit01]

The user controls content adaptation in customization. Adaptation is based on the information the user gives about himself and his preferences. Customization can affect both the content and the appearance. A user can, for example, decide the fonts and colours of the user interface. The content can be customised when the user chooses topics that interest or do not interest him.

Normal methods in hypertext adaptation are guidance, ordering and hiding. They are normally used with links. They can be used in all multiple media when the content consists of multiple smaller information components. If the content is one big component with an exact and ordered structure, this structure should not be changed. Guidance means that some recommendations are given to the user based on his user profile. The user profile is also used to sort the data. The new order should emphasise the information that is believed to interest the user most. In the same way some information that is not expected to interest the user can be made hard to access or even hidden.

Both personalization and customization enable tracking of the user data. The user has to be identified to adapt the content. When the user is identified, his behaviour can be tracked. This brings added value to the content provider. It makes it possible to charge the user based on usage. It also enables one-to-one marketing systems and personalised advertisements. [Neu00]

Legal aspects have to be considered when user tracking and profiling are used. The customer’s consent is needed to gather data and to use it for targeted advertising. [Neu00] [SNE97]

The medium and the user are not the only things that adaptation can be based on. The location of the user can be taken into account. Local newspapers have news about local events. This can be taken further when the current location of portable devices can be detected. Adaptation can also be based on the time. For example, television beer commercials are not suited for early morning, but they are broadcast in the evenings. In a more advanced version, a beer commercial would be delivered to the user's portable terminal when he is leaving his work on Friday afternoon. [Jää01]

3 Digital television

3.1 Digital television in Finland and in Europe

For a while it has been possible to receive digital television broadcasts in Europe and also in Finland. In Finland digital channels have been available for satellite dish owners since October 1998 [IDA00b]. These digital broadcasts conform to Digital Video Broadcasting, DVB, standards. DVB has been a mutual standard in Western Europe and in some other countries in the world, but other rival technologies exist, for example, in the USA.

Digital television broadcasting has been done in a uniform way in Europe. In spite of this, the user does not have the freedom to buy the receiver from the manufacturer he wishes. In addition to plain television program broadcasts, there are also so-called *value-added services*. Value-added services are applications available to customers. The most common examples of value-added services are the Electronic Program Guide, EPG, and super teletext. Value-added services have been system specific so far, which means that value-added services of the broadcast can be accessed only with a service compliant receiver that is typically provided by the broadcaster. An application designed for one system, i.e. to one broadcaster's channels, does not work on other system's receivers. Examples of these competing systems are OpenTV and Canal+ Media Highway. There are also various conditional access technologies that cause non-interoperability. A conditional access system is needed, for example, with pay services.

While digital commercial television channels have been available in Finland via satellites, public terrestrial television broadcasts have been analogue. In Finland terrestrial digital television broadcasts were introduced in the end of August 2001. An integrated receiver or a set-top-box is needed to receive the terrestrial broadcast. For public digital television broadcasts Finnish broadcasters and other related companies agreed to conform to the NorDigII standard specified by NorDig [Nor01]. NorDig is an organisation consisting of over twenty companies from Finland, Sweden, Norway, Denmark and Iceland. NorDig's target is to provide open standards and a uniform equipment market for Scandinavian countries. According to NorDigII, digital television broadcasts have to conform to DVB standards. NorDigII specifies that value-added services are based on the Multimedia Home Platform (MHP) specification. MHP is discussed in section 3.2.

Digital television's value-added services enable interactive television in a new way. Interactive applications can be downloaded and executed. Applications are normally carried in the television broadcast. Some DVB receivers can even access the Internet. Also in the analogue television broadcasts there have been interactive applications. For example, it has been possible to publish classified ads in teletext using the telephone. Also, so-called SMS chats are very popular: users can send an SMS message from their cell phone, and the message is then broadcast. However, digital television makes it possible to increase the level of interactivity, thus enabling more complicated applications if not completely novel ones.

Television is converging with other media and this will affect multiple media. The use of television in multiple media will become easier and more natural. Examples of this already exist: MHP applications for DVB terminals to access WAP and SMS services have already been developed. Even though they are just using an existing medium in digital television, this still shows the possibilities. In 2002, one in every five households in Europe watched digital television. Jupiter MMXI expects that by 2006 the proportion will increase to over 50% [Lah02]. It is also expected that most of the viewers will access interactive services [Neu00].

3.1.1 Characteristics of the television platform

The television platform has its own characteristics like any other platform. These characteristics have to be taken into account when digital television is used as a one medium in a multiple media system.

With PCs, users 'pull' information from the Internet. The user triggers the information retrieval, and the documents are requested, or pulled, from the Internet. The 'push' method is normally used with television. The broadcaster pushes information with the broadcast and the user just receives it. Only the pay-per-view type of services can be considered using pull method with television. The gathering of information for user models is hard to achieve when content is pushed to the user. The push method affects user behaviour as well. The PC is used actively, while television can be used more passively. The content should also reflect this. The PC is used to access information, for example relating to work or hobbies. Television, on the other hand, is used mainly for entertainment purposes. This could change to some degree with the interactive services of digital television, but it is hard to predict how easily the role of television will change.

While the display area of a typical television set is considerably larger than that of computer monitor, its resolution is poorer. The television set's readability is therefore relatively weak. The benefits gained from the big screen are also lost because of the long viewing distance. Even the whole resolution cannot be used, because some margins has to be left on each side. Television sets tend to overscan and leave some area on each side outside the displayed area. In addition to poor readability, the lack of input devices affects usability. Normally a plain remote controller is the only input device. This affects especially applications that require text input. [RaH00] [RiP01]

3.2 Multimedia Home Platform

"MHP defines a generic interface between interactive digital applications and terminals on which those applications execute" [MHP01]. In practice it gives the rules that digital television device manufacturers and application developers have to follow. Terminals are normally digital television sets or normal analogue sets with a set-top-box. It is anyway possible to build other kinds of terminals by following the MHP specification.

MHP tries to provide a way to move from the vertical digital television markets to horizontal ones, as seen in figure 2.

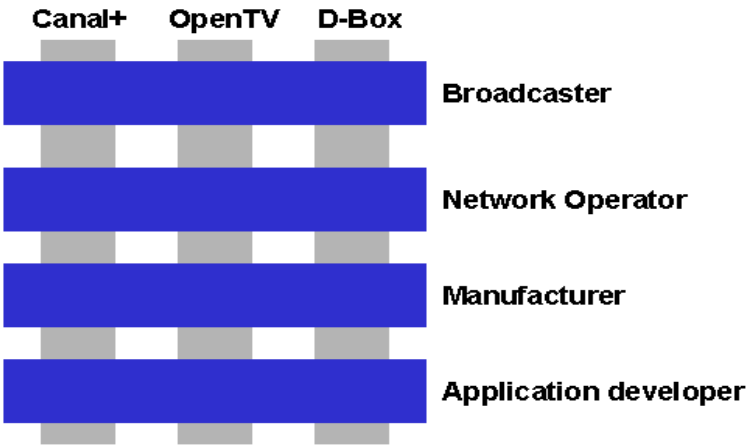

Figure 2 Transition from vertical digital television markets to horizontal markets

In vertical digital television markets the service provider operates on all levels: application development, manufacturing, network operating, and broadcasting. There are no uniform interfaces specified between the levels. It is not likely that various systems are interoperable in the vertical markets. In the vertical markets a consumer needs, for example, an OpenTV compatible receiver to access OpenTV's value-added services. The consumer cannot use a receiver from Canal+ to access them.

In horizontal digital television markets each level has a well-defined interface. Implementations from different providers can work together. Companies can target their products at one level only and they need not operate on all the levels. MHP's key element is a generic API. It enables the MHP applications from the different providers to operate on all manufacturers' hardware and software implementations. Finnish cable television companies listed lack of standards as one of the main barriers to the digitalisation of cable TV networks [IDA00a]. For terrestrial broadcasts this is now solved, because Finnish broadcasters have mutually agreed to follow the NorDigII standard. NorDig II compliant receivers support all APIs and content formats defined for the Interactive Broadcast profile of Multimedia Home Platform, MHP, version 1.1 [Nor01]. MHP is the part of the NorDigII that enables interactive applications. MHP profiles are discussed in section 3.2.3.

It should be noted that there is no requirement for DVB receiver manufacturers to support NorDigII. That means that DVB receivers sold in Finland might not support MHP at all. Consequently, the first set-top-boxes sold in Finland in August of 2001 did not support either MHP or NorDigII. As late as in November 2002, first MHP compliant set-top boxes became available. Sony released a high-end integrated MHP television set earlier, but that is clearly targeted for special groups.

3.2.1 Network connections

An MHP terminal is connected to one or two different networks, depending on the profile that the terminal belongs to [MHP01]. The profiles are discussed in section 3.2.3. An

MHP terminal is always connected to a broadcasting network, which is the television network. This connection is called the *broadcast channel*. The broadcast channel is one-way only. Data moves from the content provider to the MHP terminals. The broadcast network can be terrestrial, cable, or a satellite network [Nor01].

It is also possible for the MHP terminals to be connected to a two-way network. This connection is called either the *interaction channel* or *return channel*. The interaction channel is essential in order to send data from the client to the content provider. This enables interactive applications. The interaction channel can also be used to deliver information from the content provider to the client. This data is addressed to one single MHP terminal instead of to multiple terminals, which is the case in the broadcast channel.

The main purpose of the broadcast channel is to deliver television programs. The MHP applications are also delivered in this same network. They are multiplexed into the program flow. In addition to the applications, some common data available for all the clients can be delivered in this way. For example, it could be the data for electronic program guide or the data for super teletext. The technique used in this delivery is called a *data carousel* (figure 3). In a data carousel modules of data are transmitted in a cyclic manner. The receiver waits for the data until it is re-transmitted.

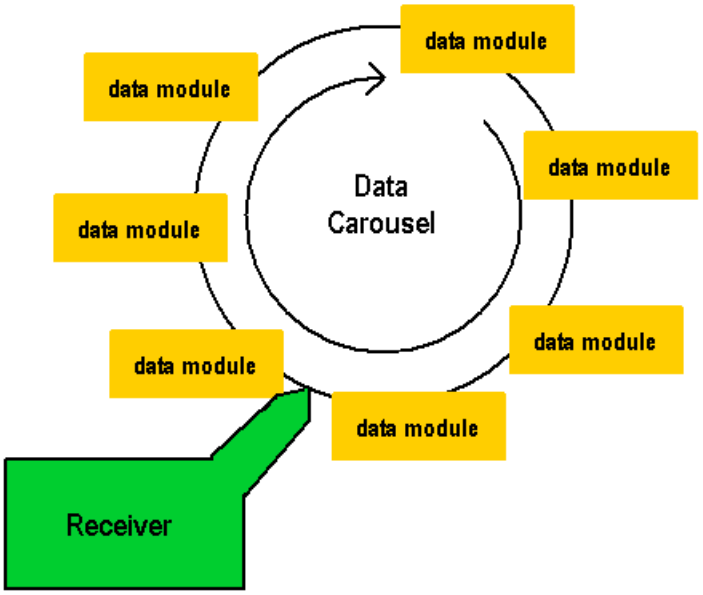

Figure 3 Data carousel.

Object carousels are based on data carousels. In the object carousels a structured group of objects is transmitted in a cyclic manner instead of more primitive data modules. The object carousels enable the transmission of file systems through the broadcast channel. File systems are transmitted as directory objects and file objects. In this way the broadcasting network can be abstracted to a file system to simplify its usage.

3.2.2 Architecture

The architecture of the MHP system consists of three layers: resources, system software and applications (figure 4) [Vog00]. The resources are logical and they are mapped to one or more hardware components of the system. The available networks, MPEG processing, I/O devices like the remote controller and the display unit, a CPU, memory and a graphics system are examples of resources. The system software provides an abstract view of the resources to the applications. The device manufacturer implements the resources and the system software, and they need to conform to the MHP specification.

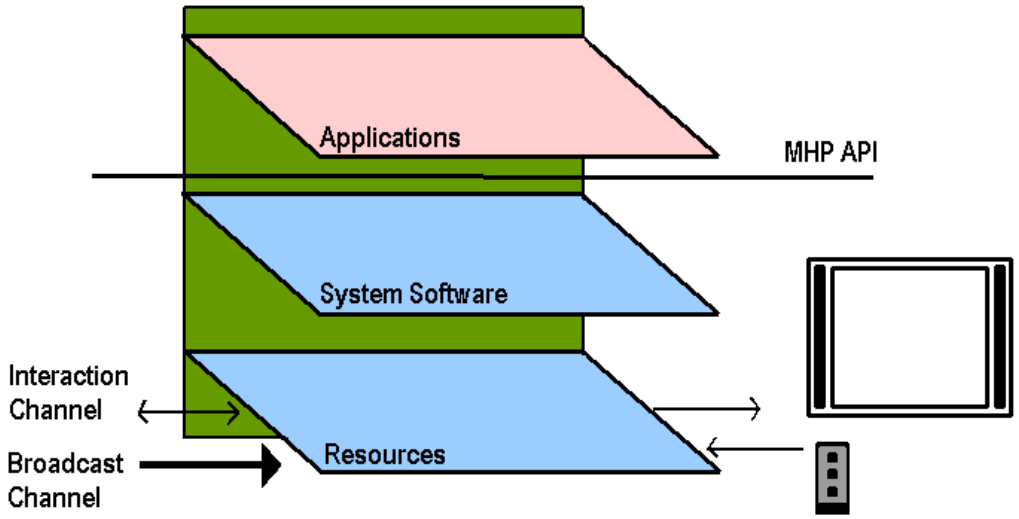

Figure 4 Basic architecture of the MHP terminal.

Applications use the system software and the resources underneath it via the MHP API. The API is implemented by the system software. The other parts of the system software are an operating system, protocols, a Java virtual machine and an application manager. The application manager is sometimes referred as a navigator. The application manager controls the MHP device and the applications running in it. Managing the applications' lifecycles is the application manager's responsibility even if the applications are interoperable.

MHP applications can be developed with Java. Java simplifies the process of beginning to develop MHP applications, since it is widely used. It is also non-platform specific. Had some new or less known programming language been chosen, the developers would have had to learn it first. Now they can focus on developing applications. *Xlet* is a generic name for Java television applications, given by Sun.

The MHP terminal has a Java virtual machine, which provides a common interface to different hardware and software implementations. The Java virtual machine is an abstract computer that runs compiled Java programs. It is generally implemented by software on top of a real hardware platform and operating system. All Java programs are compiled for the virtual machine. Therefore, the Java virtual machine must be implemented on a particular platform before compiled Java programs will run on that platform. While Java is used to develop increasingly complex applications, the earliest MHP terminals may not have enough processing power, memory, or library classes to make all MHP applications

possible. It is somewhat unfair to compare MHP terminals with PCs. It could be more appropriate to consider an MHP terminal as an embedded system, where the most important requirements are not related to performance or appearance, but to reliability. Since MHP applications are downloaded, their size is also restricted. The MHP-specific Java platform is named DVB-J. The DVB-J platform is presented in figure 5 [Vog00].

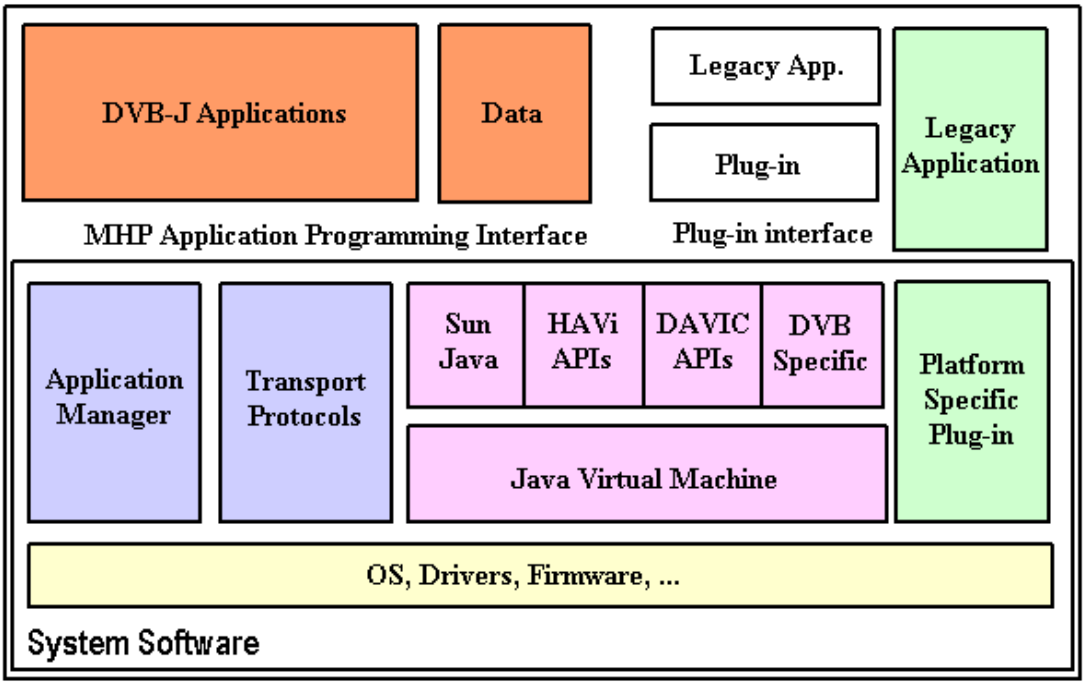

Figure 5 DVB-J Platform

In the figure a more detailed view of the system software and applications is presented. The whole software system is built on low level software like drivers and an operating system. These software components provide an interface to the resources for the higher-level system software components. An application manager, transport protocols, and a Java virtual machine are essential parts of the system software.

In addition to a Java virtual machine, some additional Java packages are present in the system software. Packages are collections of compiled Java classes. Classes belonging to the same package are normally related to the same problem or application domain. An application programming interface, API, consists of one or more packages.

In the DVB-J platform, fundamental Java APIs are imported from the standard Java platform. There are some restrictions on the use of those APIs. Additionally several APIs are provided by Sun: a subset of Java's abstract window toolkit, AWT, for user interfaces, the Java media framework, JMF, and Java TV APIs are included. In addition to AWT, user interface classes from HAVi, which are designed to be used in the TV environment, are included. There are also APIs defined by DAVIC and some DVB specific APIs. These Java components are always available in MHP terminals, and applications are being developed using them.

The system software implements the MHP API for DVB-J applications. Sometimes it is necessary to be able to execute applications that do not comply with MHP. Normally, these are legacy applications originating from other systems such as OpenTV. Proprietary plug-ins make it possible to execute them. They implement the necessary APIs. Plug-ins can be implemented either in the system software by the terminal manufacturer or as a DVB-J application. On the other hand, plug-ins make it possible to develop applications that will run only on a certain manufacturer's terminals. While this is suitable for enabling old applications, the market leader can turn its own plug-in into a de facto standard. This is against the leading concept behind the MHP specification.

The DVB-J application is one way to implement an *MHP application.* The other way is DVB-HTML. DVB-HTML is markup language similar to traditional HTML. DVB-HTML supports CSS for the definition of appearance. It also supports ECMAScripts to add functionality. It is based on XHTML and is therefore XML compliant. The DVB-HTML application is a set of DVB-HTML documents with one document as its starting point. The DVB-HTML user agent is available in some MHP terminals. A user agent is an application that interprets a content format, which in this case is DVB-HTML. Also the user agent can be a downloadable DVB-J application.

3.2.3 Profiles

MHP profiles combine a set of functionalities needed to be implemented by a MHP terminal of that profile. One profile is targeted at one application area. Currently (November 2002), those areas are enhanced broadcasting (EB), interactive broadcasting (IB) and Internet access. The profile is identified by the application area, but also with a level number. The number of application areas can change and new levels can be added to the MHP specification if new features are needed. This doesn't affect the older MHP terminals and the applications targeted at previous profiles. If additions had not been possible, terminals and applications would become outdated when new mandatory features are added to the existing profiles. MHP 1.0 introduced the profiles of first levels for all three application areas. MHP 1.1 defines second levels, and profiles in this version are enhanced broadcasting 2 (EB2), interactive broadcasting 2 (IB2), and Internet access 2 [MHP01]. An overview of the profiles is presented in figure 6.

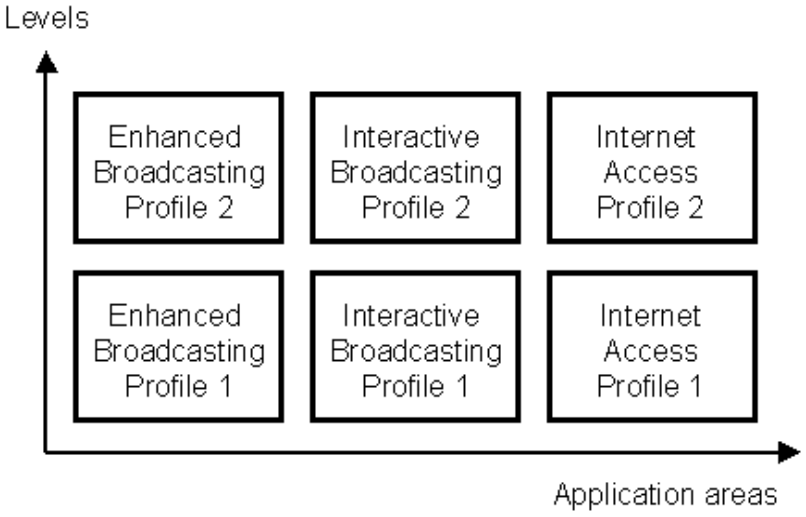

Figure 6 Application areas and levels of profiles.

The return channel is not present in the EB2 profile. The IB2 profile defines the return channel and the Internet access 2 profile has an access to the Internet. The return channel affects the content of the MHP API and the protocols that are available. NorDig II specifies that all compliant devices need to belong to IB or a more advanced profile [Nor01]. To be specific, this is IB2 because it is the IB profile specified in MHP 1.1.

The EB2 is a subset of the IB2, and the IB2 is a subset of the Internet access 2 profile. It is specified that the profile with level n is a subset of level $n+1$ of that same profile. Also the IB1 profile is a superset of the EB1 profile.

Despite the profiles and backward compatibility, developing standards will affect the television markets, where compatibility has been a tradition. The consumer can still use a thirty-year-old television set. In the future it is possible that new services will require the most recent generation of terminal. If the consumer wishes to use all possible services available, he will have to upgrade his MHP terminal repeatedly. The software of the terminals can be updated, but this might still lead to a shorter terminal lifecycle than what the consumers expect.

4 Methods to deliver content to MHP compliant digital television

Digital television aims to provide more than plain television program viewing. This means it can have potential as an additional medium in a multiple media system. There are two technologies that can be used to develop a multiple media application for the MHP compliant DVB receiver: the MHP application (either DVB-HTML or DVB-J application) or WWW pages.

If DVB-HTML is used to develop a multiple media application, development requires a generation of DVB-HTML documents, which have to be made accessible. This method is quite congruent with traditional WWW publishing.

The user can download a DVB-J application to the MHP terminal. This is one of the possibilities to use digital television as a channel in a multiple media system. The application can use any data format it wishes to use. It is not strictly tied to DVB-HTML. Developing a DVB-J application requires more work, but more advanced functionality and specialised appearance can be achieved, because functionality and appearance are not restricted by CSS and ECMAScript. The MHP terminal has to have access to both the application and the data it uses.

The problem with these two methods is the access. Applications and the application data shared by all users are delivered in the broadcasting channel within the television broadcast. The only one who can decide on the content of the broadcasting channel is the broadcaster, i.e. the television company. The broadcaster's approval is needed to put the application into the program flow. Also for the user-based adaptation, the interaction channel is needed. The interaction channel will be present in the DVB receivers in Finland if manufacturers begin to follow the NorDigII standard. It is not required though, even if the broadcasters are committed to NorDigII. The interaction channel can be used for the delivery of the adapted content, and all the data does not have to be delivered through the broadcast channel.

Applications can be downloaded from the interaction channel too. This enables third-party applications delivered to the MHP terminals. Since the TCP/IP and HTTP protocols are available in NorDigII-compliant terminals, applications can be downloaded from the Internet [Nor01] [MHP01]. The user invokes a download manually, or an application can start a download from the interaction channel.

The MHP Internet access profile terminals can access the Internet, and they have a built-in web browser. If WWW has been a medium in the multiple media system, now digital television can access the same medium. This is probably the easiest method to deliver multiple media content to an MHP terminal. The drawback is that this method leaves out the terminals that do not belong to the Internet access profiles. The characteristics of the television platform should be taken into account anyhow. If the WWW medium is targeted at PCs, the appearance on the television might be poor, and navigation might not

be suitable for a remote controller. This is why a digital television has to be considered as a separate medium. The appearance and functionality can be tailored in this dedicated medium. The publishing process resembles the normal WWW publishing process.

5 Implementation of one method

5.1 Arguments for practical testing

In the MobTV project, of which this report forms a part, mobile television applications are developed. They utilise content originally created to other media, and they can thus be seen as an additional medium in a multiple media system. The applications do not necessarily have to be MHP compliant. MHP would be a preferred choice if MHP fulfils the requirements of the project. This is what this research project intends to establish.

One of the objectives of the research is to find out how MHP can be used in practise as a medium in a multiple media system. Possible methods to use MHP terminals as a medium were introduced in chapter 4. The content can be made available to MHP terminals by converting the content to DVB-HTML documents and making them accessible. This method is quite uniform to the one used in WWW publishing. In some cases MHP terminals can access the WWW content as such. Another method is to develop a DVB-J application to be used as a user agent. This differs from typical electronic media. While the content conversion, creation and distribution phases are present, lots of effort is needed to develop the DVB-J user agent. For example, in the WWW medium content is viewed with the browser. Content providers do not need to develop a user agent. Developing a user agent does not necessarily need to be particularly complicated. It will require work, which has to be taken into account. The content providers might not be familiar with the development of this kind of applications.

The biggest challenge in using MHP as a medium is the novelty of the MHP standard. It is not clear what can be done, and how it can be done. There are not many development tools such as IDEs, which are available for almost all other platforms. There are not many books or guides either. Besides this, the television platform differs significantly from normal PC environment. This also affects the development of the software, not only the tailoring of the content. One cannot develop the software on the same platform for which it is intended. This is not especially the problem. Applications are constantly developed for cell phones and other platforms, which cannot be used as development environments. For these platforms, the software is developed and tested preliminarily on PCs using simulation and development tools. The software is then transferred to the target platform for more thorough testing. For MHP, only a few of these tools are readily available. It is possible that these tools also have to be developed.

Using MHP as a medium is therefore more risky. There are more unknown issues than in more common application development due to the novelty of the MHP specification. Developers cannot predict the forthcoming problems as reliably as needed. One way to reduce these kind of risks is to develop prototypes to find out possible problems and to clear things up. A prototype was also developed in this research. The prototype's required functionality is presented in section 5.2.1. Mostly these uncertainties affect the MHP software development in general. Firstly, the development and simulation environments needed to be set up since they were not widely available at the time (December 2001).

Also the MHP API should be tested to see how it can be used and what kind of unexpected effects it might produce. MHP terminals do not support a wide variety of content formats. This does not affect the development of the software, but it does affect the use of MHP as a medium.

5.2 Development of the prototype application

In this section the prototype development project is presented in the same order it was originally carried out. The problem of video playback is discussed in detail. Also the design of the development and the simulation environments get more attention than more trivial tasks such as the implementation phase.

5.2.1 Requirements

The prototype should make possible for users to view old television news broadcasts. At least part of the software is supposed to be executed in MHP terminal. This part should be MHP compliant. The application shows a list of broadcasts to the user, who can then browse the broadcasts with a remote controller. The user can select one news broadcast for viewing. The video clips are stored in the IMU2 database. The IMU2 system is presented in next section. The video clips are viewed in full screen, which sets requirements for their quality if they need to be converted to other formats. In addition to MHP, other established standards and methods are preferred.

IMU2 system

In the prototype, existing components from the IMU2 system are used wherever possible. These components include the database, the database interface and the existing content. IMU2 is a multiple media system developed in the Applications of Integrated Publishing project. Its media include the Internet, WAP, MP3 for the visually impaired and television. The television platform used with IMU2 is a custom set-top box, which is not MHP compliant. [Söd01]

The IMU2 system combines the content from various newspapers and television news. IMU2 is a prototype system, which had over three hundred users during the active testing period. Most of the users accessed it through the Internet medium using a PC. The content of IMU2 is divided into *channels*, which contain newspaper articles and pieces of television news. There are two kinds of channels: pre-defined and users' own. User-defined channels have customised content. The user is able to browse the content by selecting a channel or by following the links within the content.

For the television medium, a specific browser was developed for the set-top box. It should be emphasised that the application implemented in this research project is not the same application on a new platform, but a completely new piece of software.

5.2.2 System architecture

Typical electronic medium architecture

Typically, most of the digital media are based on the client/server architecture. In this architecture, the user uses a client application. The client application is located on a different system than the server application. The server application provides data, or *services*. The client requests services from the service provider, the server. The amount of data processing that is done in the client application can vary. If the client application only shows the data to the user, and all the processing is done in the server, the client is called a *thin client.*

When, for example, WWW or WAP phones are used as a medium, their clients are relatively thin. Most of the processing is done in the server. It is also typical for these media that the builder of the multiple media system does not develop the client application. Existing software, for example a web browser in a WWW medium, is utilised instead. This means that it is the content provider's responsibility to create the system for content creation, conversion and distribution. For example, the content provider converts a newspaper content to HTML documents and publishes it on the Internet. The client application, the web browser, is not developed by the publisher.

Three-tier architecture

In the prototype the IMU2 database is used and it should not be accessed directly from the client due to security issues. This constraint forces the application to follow a so-called 3-tier architecture.

A three-tier architecture is an instance of N-tier architectures. An N-tier architecture contains three or more tiers, or *layers.* A three-tier architecture is presented in figure 7.

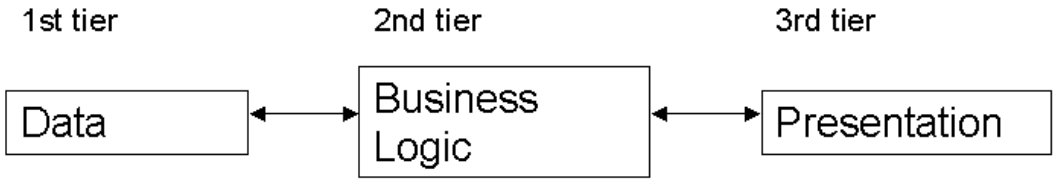

Figure 7 Three-tier architecture

The presentation layer is visible to the user. The data layer is just the data, while the functionality is in the business logic layer. Security is not the only benefit gained from the three-tier architecture as compared to the older two-tier architecture, where the presentation layer accesses the data directly. The client does not have to know anything specific about the data storage. Changes can be made to the data storage and the client can still stay the same, because the second tier offers the same services as previously. This can be applied the other way around too. Also, the business logic layer can access the data from multiple sources without affecting the client. The clients can be added without any changes to the data storage, which is important in multiple media. In addition to this, the second tier can offer more services such as queuing, caching, etc. [Cha00] [Cha01]

It is possible to divide the application into more than three tiers. The layers in a three-tier architecture are just divided into smaller parts. For example, in a five-tier architecture there could be a presentation GUI layer, a presentation logic layer, a business logic layer, a data access layer, and a data layer. The most common architecture nowadays is still the three-tier architecture. Applications rarely gain enough advantage from more than three layers. With more than three layers, the code easily becomes too complicated and separated. [Cha00][Cha01]

5.2.3 MHP in this project

To make sure that MHP compliant applications can provide the required functionality, possible problems should be identified and solved with a prototype, as discussed earlier in section 5.1. In the MobTV project, those possible problems are related to the following subcategories:

connection to a database

playback of video clips

implementation of a user interface

handling of remote controller events

a development and a simulation environment

By developing a method to deliver existing content to MHP terminals, those uncertainties can be solved. Once the software has been developed, we should know whether MHP is appropriate to be used in the MobTV project. The results can be appended to other projects and multiple media systems also, not only the MobTV project. The requirements of the prototype application were defined in a such way as to ensure that the above-mentioned problematic areas will be covered.

Problems related to implementation of a user interface and handling of remote controller events are such that they would need to be solved in any project in an organisation where MHP is used for the first time. Development and simulation environments also have to be set up in any MHP project. Those are general MHP issues. They can be solved just by setting up the environments, perhaps with some bought components. When the development and the simulation environments are ready, a simple application can be developed. Any possible problems related to the user interface and the remote controller can be identified by using them. These three problematic issues solve themselves during the project. Unknown problems, if there are any, become known problems, which can solved.

Connection to the database and the playing of video clips are not relevant to every MHP application. The server handles the connection to a database in a three-tier architecture. Some kind of communication between client and server is needed in any interactive MHP application that adopts the three-tier architecture. This problem has to be solved when the application's architecture is designed, and 5.2.5 deals with this. The required audio and video playback can be expected to cause problems. This issue is discussed in the next section.

5.2.4 Video and audio

Premises

The playback of video clips with audio is the most challenging issue in this particular project. There are four issues concerning the video clips that need clarification:

transferring data to the client

compression algorithm

decompression of video and audio

synchronisation between the video and the audio

Before the video can be played, it has to be transferred to the client. In order to do so, the clip has to be available to the client. This is easily achieved, since MHP terminals belonging to IB or more advanced profile can access data with HTTP and TCP/IP. It is adequate to make the video clips available on a web server. However, there is unfortunately no support for any streaming protocol in MHP terminals, which makes it impossible to use streaming in this project [MHP01]. The implementation of a streaming protocol during this project would be far too laborious. Before playback can begin, in practice the whole file has to be downloaded to the terminal. This should be taken into account when video compression is decided. Encoding that produces small video clips should be preferred. Currently there is no requirement for a MHP terminal to have a local storage, so this downloading method might not work with all terminals [MHP01].

MPEG-2 can almost be called the native video compression of MHP terminals: every terminal should be able to decompress MPEG-2 video streams effectively, because television broadcasts use this compression. In this project the terminal's MPEG-2 codec cannot be used for a simple reason: it is not required for an MHP terminal to be able to play video from any other source than the broadcast [MHP01]. In this project, it is one of the requirements that the video is downloaded from the interaction channel. Other video codecs are not required to be available in an MHP terminal.

Currently in IMU2, news broadcasts are encoded in ASF file format with MPEG-4 video compression [Söd01]. This is not suitable for MHP. This means that the clips have to be converted when needed, or in the worst case a totally new version has to be created each time a new video clip is added to the data storage. This important need for an appropriately chosen format was discussed in section 2.5.

MHP compliant alternatives

Video content cannot be presented in an MHP terminal from the interaction channel, unless the codecs are delivered with the player application. Otherwise the application would require more from the terminals than the specification necessitates. It should be noted though that it is not necessarily against the specification. In theory a DVB-J application can use any compression algorithm, because one can write the codec and deliver it with the application. An audio codec cannot be delivered with the application, because there are no audio-related classes available for application in the MHP terminal's Java environment. MHP terminals support MPEG-1 layer 2 or 3 audio compression.

Synchronisation also has to be taken into account. There has to be some mechanism to start the audio and the video stream playback at the same time. The players have to share the same clock as well. Also some intrastream synchronisation technique has to be used in the coded, so that the streams are played at the same rate.

MHP uses Java Media Framework, JMF, for audio and video playback [MHP01]. An actual JMF implementation is provided with the MHP terminal. The audio has to be played using the classes and the methods specified in JMF and implemented in the terminal. JMF also supports interstream synchronisation with synchronised playback start and shared clock. Fortunately, one can extend JMF to support new codecs and file formats. If an additional Java-based codec is available, it is possible to modify it to be used with JMF. This way there would be some synchronisation between the audio and the video streams.

The implementation of a video codec from scratch is not reasonable, and it would be beyond the scope of the current project. Some existing codec has to be used. Java codecs have been developed in various projects. For example MPEG-1 compressed video have been played with a reasonable frame rate and resolution, but the Java codec was not freely available [TBM99]. In addition, they do not seem to have been used with audio. Furthermore, they are usually low bitrate codecs such as H.263. Their quality would not be good enough. The difference in quality between these video clips and the digital television broadcast would be too noticeable. Even if an existing codec is available, its source code has to be also available or some other MHP compliance validation method has to be used.

While JMF is just a framework, Sun offers a couple of implementations for free. The pure Java reference implementation of JMF offers some codecs, which are pure Java. Unfortunately those codecs are not included in the MHP terminal anyhow. However some JMF implementation has to be included in the simulation environment. If the pure Java codecs are already installed in the simulation system, they can be considered almost as satisfactory an alternative as the Java codecs delivered with the application. If the prototype is ever converted to be used in actual MHP terminals, then the codecs have to be delivered with the application. For this prototype, this alternative could be acceptable although it is not MHP compliant. The reference implementation has for example Cinepak and H.263 codecs. Again, they are meant to be used with low bitrates and do not suit the needs of this project, for which better video quality is needed.

It should be kept in mind that the MHP terminals will not likely be as powerful as typical PCs, and video encoding, especially with Java, is very wearisome. The result is that video decompression can not be done efficiently, and the application's usability will become poor.

While the application is targeted at the future, and currently no preferred alternative can be found, the future should be predicted. If there is ever a desire to play video content from any source other than a broadcast channel, the specification has to be refined. JMF has been chosen for the MHP platform most likely because it can be extended with new codecs without affecting the interface. This assumption, that JMF will remain and new codecs will be added, can be used in the decision regarding the solution to the problem of video playback.

Best suitable non-MHP compliant solution

Logically, the used codec, whether its pure Java or not, does not affect the rest of the application. Java-based video decompression with MPEG-1 audio would make the application MHP compliant in theory, but the prototype nature of the application and the limited time available have to lead to a simple approach. The MobTV project group decided that the prototype application may be developed using any available JMF implementation. The upgrading to possible future extensions to codecs supported by MHP or pure Java codec is straightforward when existing framework, versatile JMF, is used.

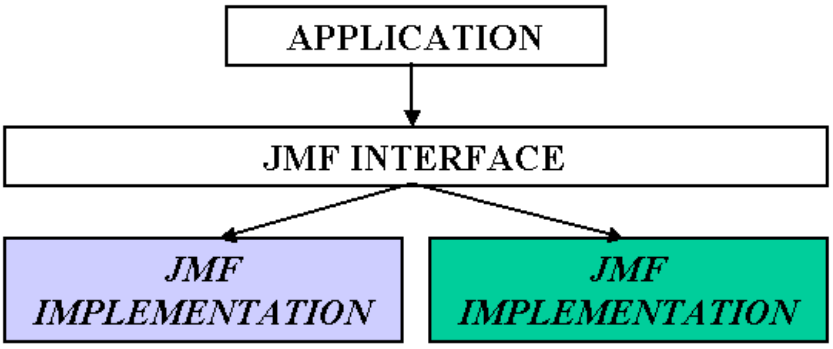

Figure 8 JMF architecture

In figure 8 the JMF architecture is presented. The application always uses the constant JMF interface. The interface can use whatever underlying JMF implementation it wishes to. This is transparent to the application. A JMF implementation is either capable of playing the video clip or not. It is possible, or even probable, that the application is not even aware of the used video compression. The JMF interface is always the same, no matter what compression is used. By doing this, the developed application does not need to be modified when the MHP terminal begins to support the chosen video compression. It is the responsibility of the database or the middle tier to offer files using the correct compression.

MPEG-2 would be the preferred compression algorithm, because the support for the compression is mandatory in all MHP terminals. Only the source is different in this case. MPEG-2 provides full DVB video quality. Therefore MPEG-2 video clips are rather big. The average bit rate of a broadcast quality video is about 5 Mb/s, which means that 5 minutes of video requires nearly 200 MB of storage. For this project, MPEG-2 should be discarded since the videos are stored using MPEG-4 codec in the IMU2 system. Previously archived video clips are needed, so the digitalisation with MPEG-2 compression cannot be added to the system. Conversion to MPEG-2 from MPEG-4 would not be sensible. The quality of an MPEG-4 video clip, which is digitised from analogue television broadcast, is not nearly as good as the quality of a DVB quality MPEG-2 video clip. The quality of video converted from MPEG-4 would not correspond to normal MPEG-2 quality, but the size of the file would increase dramatically, because MPEG-2 compression does not compress the data as effectively as the MPEG-4 compression algorithm.

Besides the pure Java implementation, Sun also offers JMF implementation optimised for Windows. It offers more codecs, but this implementation works only with Windows. Microsoft's MPEG-4 codec was used in the IMU2 project, and this codec can be used with the Windows version of JMF. JMF implementation needs AVI files instead of the ASF files used in IMU2, so a conversion between these file formats is needed. This could be the method to use the IMU2 videos in this prototype. At least Microsoft's DirectX GraphEdit tool can be used in the conversion from ASF to AVI.

There is one possibility left that would not require any format conversions. In the IMU2 system, television news clips are analysed. This analysis takes MPEG-1 format as an input. Normally, this intermediate format is not available for IMU2 clients. They can easily be made available, and Windows optimised implementation of JMF can however play them. In this manner, these video and audio problems are somewhat solved. Eventually this solution was chosen. It is as good as any other alternative when compatibility is considered, but it has an advantage: Windows JMF implementation is capable of streaming MPEG-1 over HTTP. The file does not have to be downloaded to the client before the playback can begin. No additional streaming protocols are needed and this way the prototype's usability becomes better.

As far as MHP implementation is considered, a code that downloads the clip to the terminal would not be difficult to develop, so long as there is a local hard disc or enough memory for the clip. It is also very probable that if MHP begins to support video playback from the interaction channel, streaming will also be included. The chosen solution is not MHP compliant. However, no alternative would have been. Because JMF is used, the application does not need much modification, if any at all, when proper codecs and either streaming or local hard discs become mandatory in MHP terminals. This is discussed more extensively in section 5.3.

5.2.5 Overall architecture of the application

Before detailed designing, it has to be decided whether the application will be a traditional client/server application or a distributed object application. The existing infrastructure needs to be taken into account. In this case, JRun software already exists on the server. The client/server architecture is commonly used and easy to understand for most of the developers. If the communication between the client and the server is done in a textual format, the client and the server do not have to be strictly tied to each other. The client and the server can be independently modified. They can even be implemented in a different language. Also, more extensive selection of clients can use the server due the platform-independence of exchanged data, thus making the server gain more from the synergy and better suited to multiple media systems.

XML could be an appropriate language to be used in the communication between the client and the server. It would provide the most generic solution. XML is clearly becoming the standard of the future and it is also used in MHP (DVB-HTML). XML can be used in almost any platform, since it is textual. It also contains metadata, which makes the interpretation of the data easier. The most critical component is an XML parser, which is not a problem in the server since several free XML parsers are available. Some simple parser is needed in the client too and it has to be MHP compliant. It is possible that in the future a general parser will become available for applications in the terminal. Currently,

parser-like features are related to DVB-HTML only [MHP01]. XML files can be processed even without a parser if the documents are simple. In the prototype there is communication between the client and the server only when the application is started and the client requests the list of video clips from the server. In this case, XML can be considered as an overkill.

In IMU2, a version of RMI with HTTP tunnelling was used. The client makes an servlet request, and the server returns a Java object in text form as a response. Java offers technology called serialisation for storing objects in a textual form. This solution is quick to develop because the servlet engine already exists on the server. It is straight-forward and suits the needs of this project. Server design is discussed in the next section, and client architecture is presented in section 5.2.7.

5.2.6 Server design

With Java, there are still numerous alternatives to develop the application's middle tier. For bigger applications, which are not only prototypes, Java 2 Enterprise Edition (J2EE) technology could be appropriate. J2EE utilises servlets, Java server pages, and enterprise Java beans. Servlets are Java applications meant to be executed on a server. Servlets respond to the client's requests. Java server pages, JSP, are documents in a target format (e.g. HTML or XML) with scripts. Enterprise Java beans on the contrary are components that follow the JavaBean component architecture. [Sun01a]

It should be kept in mind that the application is a prototype. And more precisely, the prototype's goal is to test MHP, not server technologies with a MHP client. The requirements for the middle tier are rather minor. Therefore the middle tier is kept as simple as possible.

There are two architecture templates for Java servlet applications. They are normally referred as model 1 and model 2. There is no major difference between these two. With model 1, the same JSP document or a servlet handles both the request and the response. With model 2, the handling of the request and the response are separated. The request is normally handled by a servlet, and the response is handled by a JSP document. In both models, database access is handled apart from the handling of the request and the response.

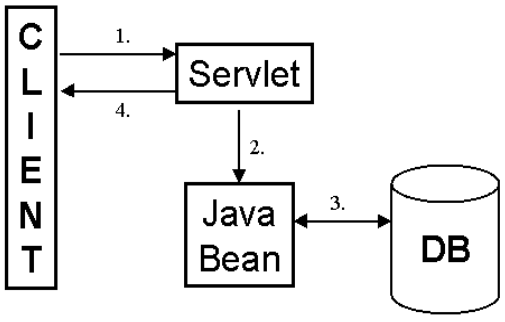

Figure 9 Model 1 Java client-server architecture

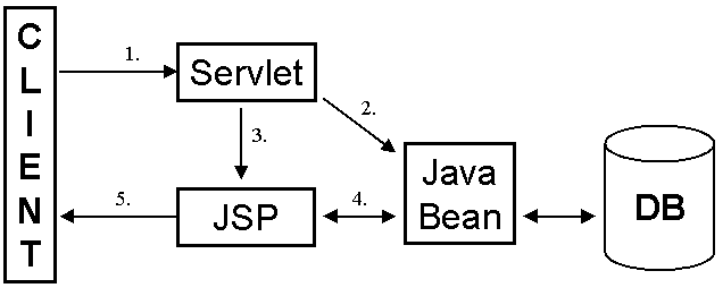

Figure 10 Model 2 Java client-server architecture

Model 1 and model 2 architectures are presented in figure 9 and figure 10 [Ses99]. In both models, all document requests are handled by a servlet, which instantiates Java beans that can access the database. After accessing the data, the control flow moves back to the servlet in model 1 and to the JSP document in model 2. The output is generated from the content of the Java bean. Finally the output is then sent to the client as a response to the request. The server architecture of the prototype follows the model 1 architecture, since the returned data is a serialised Java object and does not have a visual appearance, which needs to be separated from the content.

5.2.7 Client design

Model-View-Controller

The most fundamental client design decision is to follow the Model-View-Controller (MVC) design. With MVC, the data (model), the user interface (view), and the logic (controller) are separated. Figure 11 presents the MVC architecture [Sun01b].

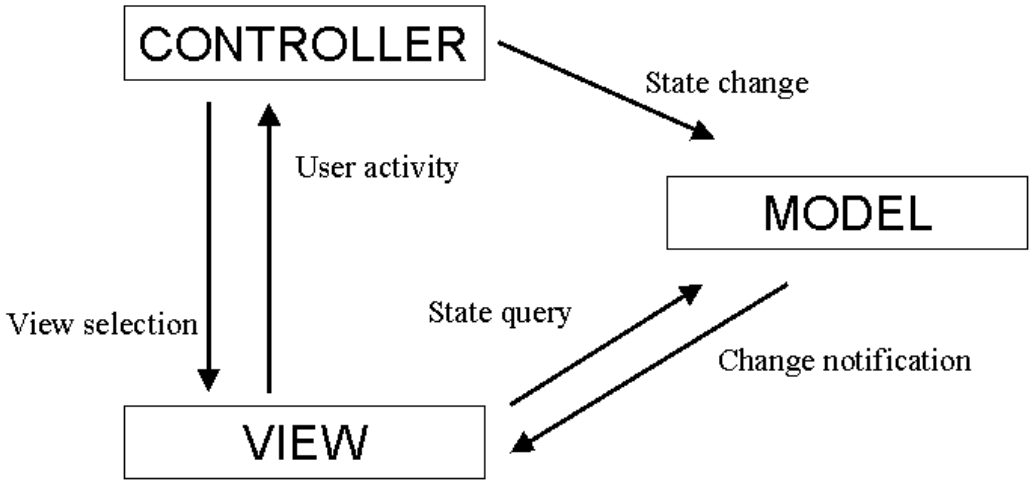

Figure 11 Model-View-Controller

The view shows the model. If the model changes, for example some data field gets a new value, the model notifies all the views tied to it of the changed value. The controller handles the user events. It can change the appearance directly by calling the view, or indirectly by changing the model. The responsibility of the view is to pass the user actions to the controller. For example, the view notifies the controller that the user has clicked a user interface component with a mouse. It should be noted that the MVC can be applied to the whole client/server application, in which case the client is just the view, while the model and the controller reside on the server. This time it is applied to the client application only.

The Model-View-Controller architecture has received some criticism, as it is claimed to be against the object-oriented design [Hol99]. According to the critic view, every object should offer its own user interface. Also objects should not be used as simple data containers, or structs, with accessors (i.e. get and set methods) for each attribute. This is easily the case with the model objects in the MVC architecture. It is said that the MVC architecture is adequate for small components, such as user interface components, but not for complete applications. In the end MVC does not meet its goal, which is a separation of the data, its appearance, and the application logic. For example, if you change the model class' attribute from integer to float, you have to change the interface and therefore also the user interface. There is still strict coupling between the user interface and the data.

Despite such criticism, MVC is still widely used. It also fits well with this specific application for several reasons. If MVC architecture is used, two of the subjects that needed clarification as to the goals of the prototype, i.e. reaction to the remote controller events and the user interface, are both in separate locations. Events are handled at the controller and user interface is implemented in the model. Also, the remote controller as an input device makes it harder for each object to provide its own user interface and react to the user input, which is targeted to those user interface components. The application's remote controller listener has to be implemented as a class of its own, which directs the events to the correct components. There is nothing wrong with this, but why not make the class controller if it has to be there. Java offers the `Observer` interface and the `Observable` class to be used in the MVC architecture.

5.2.8 Development and run-time environments

MHP is a new standard, which can be clearly seen in the MHP development tool market. There are only a few development tools available. A couple of them, such as Alticast's AltiComposer (www.alticast.com) and Snap2's MHP Express (www.snaptwo.com), are only meant for developing applications without an interaction channel, and basically only offer some GUI components. Both of those tools were tested, but they were not appropriate for the research since they were more or less addressing the GUI, which is not a major problem in this particular research.

There is at least one development implementation of an MHP terminal that is available: MHP SDK from Philips (www.mhp.philips.com). Also, Institut für Ründfunktechnik (IRT) offers its MHP reference implementation for licensing and use in the application development (www.irt.de). Both are fairly expensive when compared to the needs of this project, so no commercial development or run-time environment was used.

The number of MHP development tools is increasing constantly. At least both Philips and Alticast have introduced more tools since the decision on the development and the run-time environments was made.

In this project the development and the run-time environments were built using free software components. Java runtime environment (JRE) from Java SDK version 1.3.1 was used as a basis. JRE consists of virtual machine and core APIs. Windows version of JMF 2.1.1 was used for the video and audio playback. MHP specification refers to an older version (JMF 1.0) but the basic interface is not changed. Implementations of older JMF versions are no longer available from Sun. JMF required the use of the standard JRE instead of the PersonalJava runtime environment. The Sun specifications for DVB, which are referred to in [MHP01], contain the PersonalJava runtime environment instead of the standard Java runtime environment. PersonalJava is intended to networked consumer devices instead of PCs. The Java TV package was also included, though only its `Xlet` interface was utilised. JMF and Java TV are both included in the Sun specification for DVB.

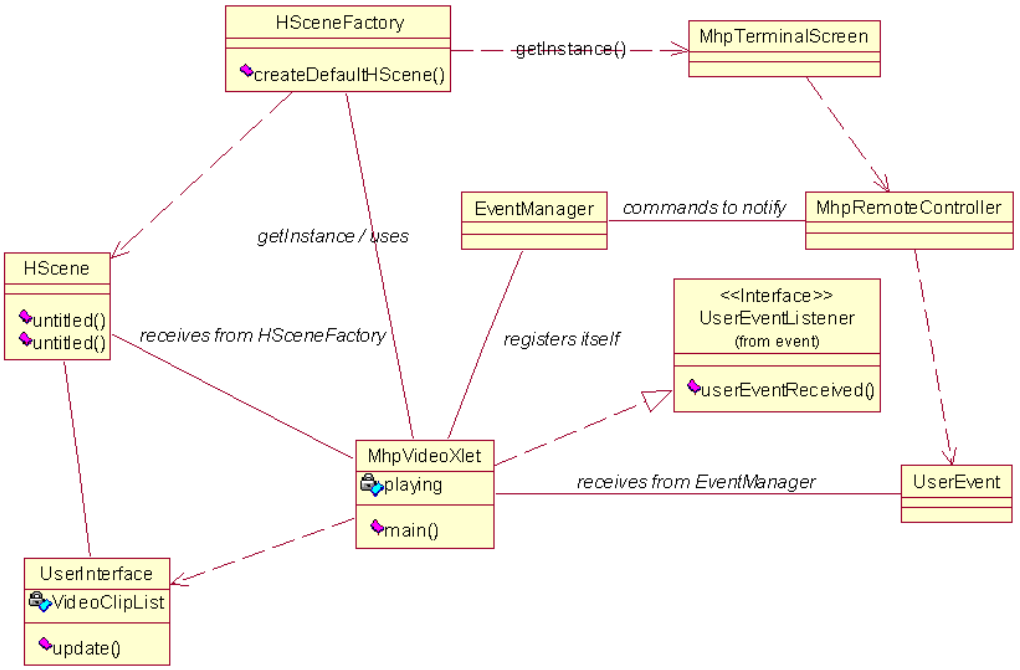

Figure 13 UML class diagram of simulation environment

Other DVB-J's software components used by the application were developed in the research. In Figure 13 the simulation environment is presented as a class diagram. Most of the classes related to this particular application and to the video presentation are left out. For user interaction some classes from the package `dvb.org.event` were developed. The application uses these classes to acquire user events from the remote controller. Constants from the class `java.awt.event.KeyEvent` were used for the remote controller key codes. Because there were not constants for the coloured remote controller buttons, the codes for function keys F20-F23 were used since they are not available on a normal keyboard. Using the same constants for the keyboard keys and the

remote controller buttons could have made the application perform incorrectly on some platforms. The method `notifyListeners` was added to `EventManager` so that the remote controller simulation class `MhpRemoteController` can easily inform registered `EventListener` instances. Classes were not thoroughly implemented and only the functionality needed by this application was implemented.

HAVi's GUI components should be available on a DVB-J platform. The free NIST DASE development environment contains the HAVi classes, but they are not thoroughly implemented (www.dase.nist.gov). However, the user interface can be built using the available set of `java.awt` package's components. One can make a subclass of `java.awt.Component` and draw the component with Java's graphic methods. However, only a few HAVi classes from the package `org.havi.ui` are essential: `HSceneTemplate`, `HScene` and `HSceneFactory`. `HScene` is a base container for a DVB-J application. The simulation environment's `HScene` class is inherited from the class `java.awt.Window`. It is not completely implemented: for example, components cannot specify their location on the y-axis. `HsceneTemplate` is used to address attributes of a `HScene`, and `HSceneFactory` is used to create `HScene` instances.

Some classes which are not a part of the DVB-J platform or part of the application were also needed: `MhpTerminalScreen`, which simulates a TV screen and is a subclass of `java.awt.Frame`. `HSceneFactory` and `HScene` are developed so that `HScene` can only be over `MhpTerminalScreen`. Only one `MhpTerminalScreen` per virtual engine is permitted. This was achieved using the singleton pattern [Gra98]. When the first `HScene` is created, the `MhpTerminalScreen` instance shared by all applications is also created. As a side effect of the construction of `MhpTerminalScreen`, `MhpRemoteController` window is also created. It simulates the remote controller. The user can click the buttons of the remote controller with a mouse and use the application as he would normally do with an MHP terminal and a television set. `MhpRemoteController` changes events generated by mouse clicks to `UserEvent` objects, which are sent to all registered `UserEventListener` instances via `EventManager`.

Because it is not required for the MHP platform to provide a GUI component for a JMF player, JMF classes also needed some modification. One cannot override classes easily. If there are two classes with exactly the same name, also the package being the same, the decision as to which class is used is based on the order of the libraries included in the classpath environment variable. This could easily lead to problems. Therefore a few additional classes were created to be used instead of the original JMF classes.

The class `MhpManager` creates an instance of `MhpPlayer`. This class `MhpManager` should be used instead of class `javax.media.jmf.Manager`. `MhpPlayer` implements the interface `javax.media.jmf.Player`, but is capable only of playing video in the background, that is in the `MhpTerminalScreen` window in this case. Because extending `javax.media.jmf.Player` is impossible, `MhpPlayer` uses `Player`'s functionality using a delegation pattern [Gra98]. In figure 14, classes related to video presentation are presented as a UML class diagram.

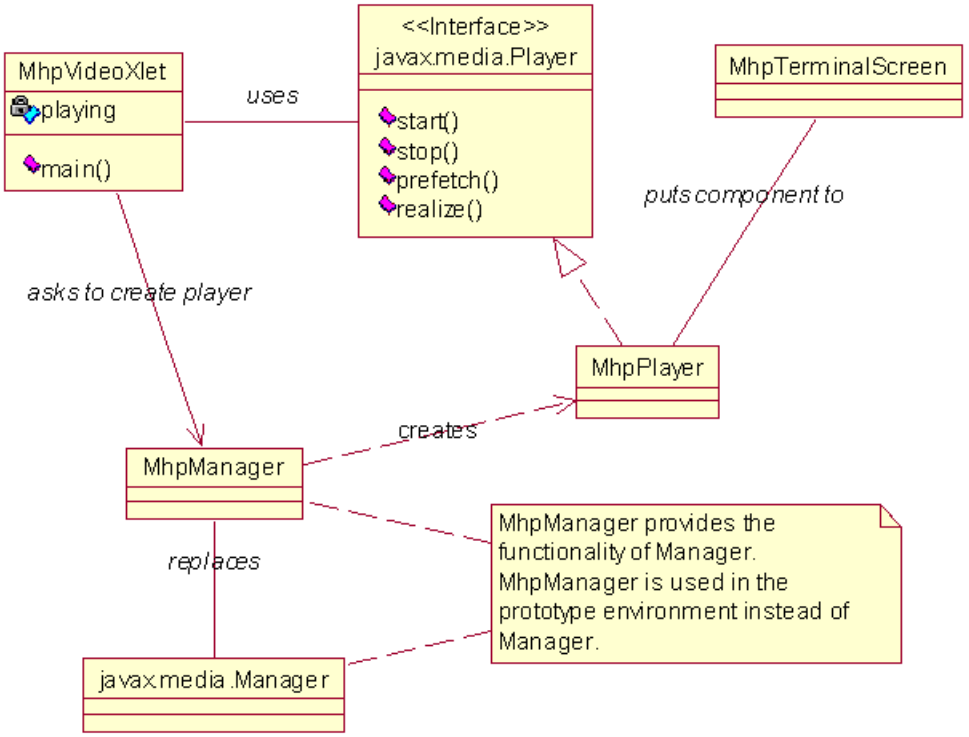

Figure 14 UML class diagram of video presentation classes

Sun's reference implementation of Java TV could have been used in the simulation environment. In this prototype only one interface from Java TV API was utilised. It was easier to develop the simulation environment without implementation of the complete Java TV API. It might be a preferable basis for simulation environment if a more broadcast-like simulation environment is beneficial, or the application needs to access for example service information.

This developed environment lacks some features that might be useful. The delivery of the application to terminals cannot be tested and this should be tested when the first terminals with an interaction channel become available. The user interface cannot be partly transparent in this environment. This feature of real MHP terminals would have been very hard to implement.

5.2.9 Implementation

The Java SDK version 1.3.1 was used in the development. If some classes were introduced in the Java version 2 (Java SDK's 1.2 and greater) and are not explicitly mentioned in the MHP specification to be available, they were not used to ensure that the application can be run in systems based on Java version 1.1.

Only one GUI component, `MhpLabel`, was implemented for the user interface. It extends `java.awt.Component` and resembles `java.awt.Label`.

Though the application was developed with a more recent Java version, it was also tested with the older JRE version 1.1.8. No errors were found. Also the PersonalJava runtime environment was tested. Only the video playback did not work, but this belongs to the simulation environment and does not affect the compatibility of the prototype application developed.

The developed application and the visible test environment can be seen in figures 15 and 16. In the first one, the list of news broadcasts is shown to the user, and in the second one the news clip is played.

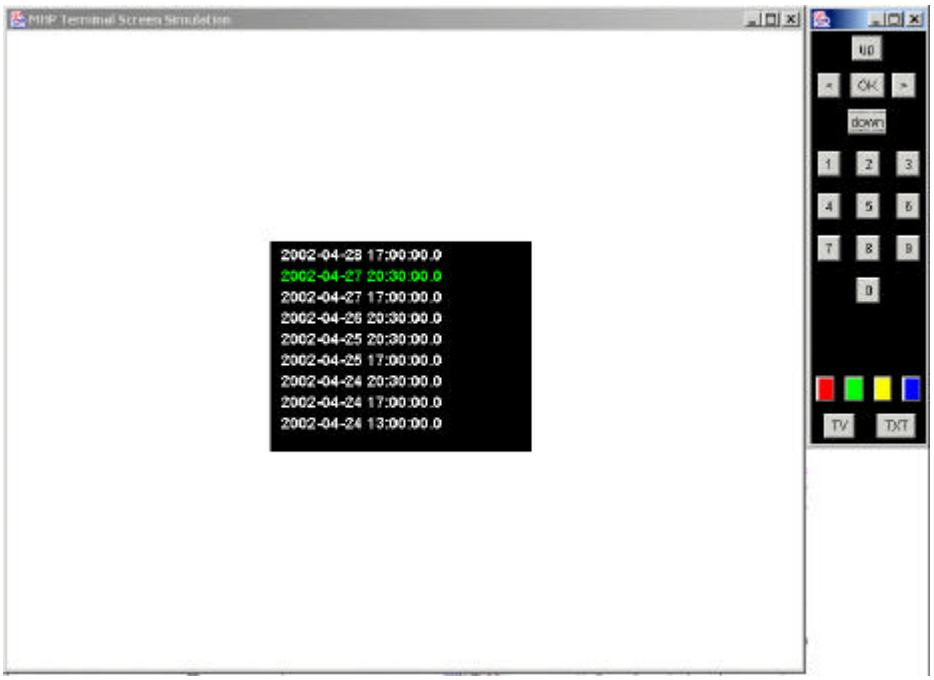

Figure 15 List of news broadcasts is shown to the user

Figure 16 News clip is played

5.2.10 MHP compliance validation

At the time of writing (November 2002), first MHP terminals have become available. MHP terminals with an interaction channel are even further in the future. Also, no validation tools exist for MHP applications. The systems of Alticast, Philips and IRT may be suitable for the validation of the MHP compliance of the developed application. They are, however, too expensive. The generated code was only validated by comparing it to the MHP specification. The specification, however, is very broad, with over one thousand pages with references to other specifications. Some issues are not yet well defined and the specification is occasionally quite confusing. As a result, it is impossible to say whether the developed application is fully MHP compliant. For example, if a mistake has been made in the interpretation of the functionality of the obligatory HAVi components, it can affect the whole application, thus making it seem completely non-compliant with MHP.

When the commercial MHP terminals become available, applications can be tested and if necessary modified for use with the particular MHP terminal. This policy was introduced because the specification may be subject to a change before the first MHP terminals with an interaction channel become available. Real applications have to be developed for real terminals. An MHP-compliant application is useless if the terminals are not compliant, and the application cannot therefore be used.

5.3 Further development

The prototype proved that the method is applicable, and therefore direct development of the prototype application can cease. The project group decided that the applications of the MobTV project will be developed to be as MHP-compliant as possible. This decision was based on the fact that this research did not find any overwhelming problems other than video playback. Although the prototype application is no longer being developed, some techniques and solutions used in it will be used in the applications of the MobTV project.

No cost-effective and sensible solution for the problem of video playback with MHP was found. A Windows-optimised JMF implementation was used for the video and audio playback, even though it is not MHP-compliant. This approach will be used in the MobTV project as well.

6 Discussion

6.1 Evaluation of the research project

The objectives of the research were met. One method to deliver content to MHP terminals was tested in practice with a prototype application. Unfortunately, the prototype application was not MHP-compliant, but no problems other than those related to video playback were encountered. The possible solutions or workarounds to this problem were studied in depth. No entirely satisfactory solution was found, but the developed prototype application can be easily modified when the required codecs become available. This solution can be recommended for other similar projects as well, if MPEG-2 video cannot be delivered in the broadcasting channel, but good video quality is needed. If low-quality video is acceptable, a Java-based codec can be developed. This way it should be possible to make the application MHP-compliant.

It was noted that using a new technology takes significantly more time than using a familiar one. When developers are not used to the technology employed, it is not clear what can be done, and how it can be done.

The novelty of the MHP specification puts more stress on this issue. There are only a few tools and guides available, and even their compatibility is unclear. The specification is easily the only thing to lean on, but it may also contain inconsistencies. This was known in advance, however, and it was one of the main motivations for this research. The aim was to test MHP and its suitability for the MobTV project in practice and to solve emerging problems before the time-critical development in this project begins. However, this is not a MHP-specific problem but a common feature of any new technology. Once MHP is used more widely, tools and guides will become available and even the specification might be refined to a level where compatibility between all the elements is ensured.

The prototype development brought a deeper understanding of MHP and its use from the multiple media's viewpoint and as a whole. The results of the research will be used in the MobTV project, of which this research project was part.

6.2 Developing a DVB-J application

Developing a DVB-J application does not significantly differ from any other application development. It uses a common programming language, Java, and rather high level interfaces are offered for the DVB-J platform. For some developers, the fact that the application is developed on a totally different platform than it is aimed at may be unfamiliar.

MHP sets some restrictions for the applications and not even all the core Java classes are available for the DVB-J applications. Usually, core Java, as its name implies, is used as a basis. It is normally extended with various packages from various sources to bring more features that can be used in different application areas. This is done with the DVB-J platform also. The subset of core Java is extended with additional packages, such as JMF, from other contexts or sources, and also with completely new DVB specific packages. With MHP the developer cannot import additional packages, as is normally done in the PC environment. There are not necessarily any guarantees that the available additional packages are MHP-compliant. Also, the fact that every additional class has to be delivered to the client has to be taken into account. While the DVB stream can contain lots of data, it will however make the application start with a delay.

On the other hand, the fact that developers are used to utilising additional packages in addition to core Java can help the adaptation to the MHP platform. The developed prototype application used only a few DVB-specific Java classes, but there are plenty available. If these are needed, they need to be studied first. This is something that most of the Java developers are used to.

A big problem in the software development could be the fact that GUI components were not available. The absence of the HAVi UI implementations is due to the novelty of the MHP specification. This will most certainly change in the future and the HAVi UI component implementations will become more widely available. MHP supports only a subset of Java's Abstract Window Toolkit, which is normally used to build GUIs with Java. This subset can be used as a skeleton of the application's user interface. Making custom UI components by drawing them using graphic primitives is not very complicated, at least when the components are fairly simple. If the application has a complicated user interface, some of the MHP development tools available should be considered.

6.3 MHP in multiple media

Digital television can be used in multiple media. Once the MHP terminals begin to have Internet access, or in some cases interaction channel is adequate, content can be offered to the MHP terminals rather easily. The available methods are presented in chapter 4. Problems should not arise with text and image content types. The content is either converted to DVB-HTML or a specific user agent for the chosen content type is developed. As noted in this project, and summed up in section 6.2, this is not particularly hard. Or at least, MHP is not responsible for additional problems or workload, especially so when the development and the simulation environments are available. It should be noted that developing an user agent is not necessarily a simple task. Even an user agent with limited functionality can be rather laborious to develop. These kinds of applications are already developed. On 5th April 2002, an application developed by Ortikon Interactive (www.ortikon.com) that offers content of Etelä-Suomen Sanomat to MHP terminals was announced.

With audio and video content types, the subject is not that clear as seen with the prototype application. A totally MHP-compliant solution was not found. There is no requirement for the MHP terminal to offer video playback from any source other than the television broadcast stream. Workaround solutions are available, but they all have their setbacks.

This was reported more extensively in section 5.2.4. Currently, MHP is not an appropriate medium if the content is video. Or to be more specific, with the current MHP specification, MHP is not an appropriate medium. There are only MHP terminals with limited functionality available yet. Even when the first NorDigII compliant ones become available, they will not be suitable for video content if they follow the current specification. If MHP becomes popular, this situation will likely change in the future. Playback of the video from different sources using video compression algorithms other than MPEG-2 should be added to the requirements of the MHP terminals belonging to more advanced profiles. Once this has happened, MHP can be used as a medium even with video content. It should be emphasised that if possible, the decision about the video compression of the multiple media system should be made after the compression supported by the MHP terminals is known. The conversion between the video compression algorithms can become the obstacle in using MHP as a medium.

7 Conclusions

In multiple media, the same content is delivered to more than one medium. Digitalisation might change television's role, and television could become a more important medium in multiple media systems. MHP is a standard Finnish broadcasters are committed to. MHP-compliant digital television receivers or set-top boxes could be used as a medium in multiple media systems. This was researched and reported in this document. Various methods to use MHP in multiple media were presented, and one method was tested in practice with a prototype application.

MHP seemed to be appropriate for multiple media systems, depending on the content type. With video, MHP proved to be somewhat problematic. A totally MHP-compliant and cost-effective solution was not found. A Java-based codec could be used, but it will not offer good quality because video decompression with Java requires a lot of computation power. The development of a Java video codec is also a very complicated task. The solution used in the development of a prototype application is easy to modify if the MHP specification changes in the assumed direction. Also, the required parts of the PC simulation environment were developed.

It was found that at least simple applications are easy to develop to an MHP platform. The novelty of the specification leads to a lack of affordable tools, and even to a lack of implementations of GUI components. This means that the MHP application development easily requires a bit more time and work than would usually be the case. This should change in the future though, and the development of DVB-J applications will become easier.

References

[Agn97] Agnew, R., Paper and pixels: automated publishing through multiple channels. Design Management Journal, Fall 1997. pp. 47-52.

[Ame00] The American Heritage Dictionary of the English language. 4th edition, Houghton Mifflin Company, 2000. [Also: http://www.dictionary.com]

[Cha00] Chaffee, A., One, two, three, or n tiers? Java World, January, 2000, http://www.javaworld.com/javaworld/jw-01-2000/jw-01-ssj-tiers.html. [3.4.2002]

[Cha01] Chartier, R. Application architecture: An N-tier approach, part 1. October 23, 2001, http://www.15seconds.com/Issue/011023.htm. [3.4.2002]

[FS-1037C] Telecommunications: glossary of telecommunication terms, Federal standard 1037C, August 7, 1996.

[Gra98] Grand, M., Patterns in Java, volume 1. Wiley Computer Publishing, USA, 1998.

[HeH00] Heise, D., Hornstein, J., The digital content challenge, a create once, publish everywhere strategy. COPE white paper, Spring 2000.

[Hol99] Holub, A., Building user interfaces for object-oriented systems, Part 1. Java World, July 1999, http://www.javaworld.com/jw-07-1999/jw-07-toolbox.html. [3.4.2002]

[HPN00] Han, R., Perret, V., Naghshineh, M., WebSplitter: a unified XML framework for multi-device collaborative web browsing. Proc. ACM 2000 Conference on Computer Supported Cooperative Work, Philadelphia, USA, December 2-6, 2000, pp. 221 – 230.

[IDA00a] Development of digital TV in Europe, 1999 report, Finland. IDATE, Institut de l'audiovisuel et des télécommunications en Europe, January 2000.

[IDA00b] Development of digital TV in Europe, 2000 report, Finland. IDATE, Institut de l'audiovisuel et des télécommunications en Europe, December 2000.

[Jää01] Jääskeläinen, K., Viekö DOCOMO koko potin? GT-lehti, 1, February 2001, pp. 6-8.

[Kit97] Kitamura, H., New algorithms and techniques for well-synchronized audio and video streams communications. Proc. Sixth International Conference on Computer communications and Networks (ICCCN '97), Las Vegas, USA, September 22-25, 1997, pp. 214-219.

[Lah02] Lahdensivu, M., Jupiter ennustaa digitelevisiolle kasvupyrähdystä. Digi Today, March 12, 2002,

http://www.digitoday.fi/digi98fi.nsf/pub/md20020312111227_mls_15842069. [7.4.2002]

[MHP01] ETSI TS 102 812 V1.1.1 (2001-11); Digital Video Broadcasting (DVB); Multimedia Home Platform (MHP) specification 1.1. November, 2001. [Also: www.etsi.org, 3.4.2002]

[Mob01] Mobile television in fourth generation networks – project's homepage, http://www.vtt.fi/tte/mobtv/. [3.4.2002]

[Mos98] Moss, K., Java servlets. McGraw-Hill, New York, 1998.

[Mou01] Moulding, J., Content management: the opportunities and costs. Cable & Satellite, issue July-August, 2001, p. 39.

[Neu00] Neunart, B., Cross Media Content. IST Consultation Meeting, Report (draft), Luxembourg, April 27, 2000.

[Nor01] NorDig II Digital integrated receiver decoder draft specification, version 1.0. NorDig, June 13, 2001. [Also: www.nordig.org, 3.4.2002]

[RaH00] Rawolle, J., Hess, T., New digital media and devices, an analysis for the media industry. The International Journal on Media Management, volume 2, issue 2, 2000, pp. 89-99.

[RiP01] Rinnetmäki, M., Pöyhtäri, A., Digi-TV:n palveluntekijän opas. Tekes, Helsinki, 2001.

[Rit01] Ritz, T., Personalised information services – an electronic information commodity and its production. Proc. ICCC/IFIP 5th Conference on Electronic Publishing, Canterbury, United Kingdom, July 5-7, 2001, pp. 48-58.

[Ses99] Seshadri, G., Understanding JavaServer Pages Model 2 architecture. Java World, December 1999, http://www.javaworld.com/javaworld/jw-12-1999/jw-12-ssj-jspmvc.html. [3.4.2002]

[SNE97] Sabelström, K., Nordqvist, S., Enlund, N., Synergy in integrated publishing of printed and electronic newspapers. IARIGAI 24th International Research Conference, London, September 1997.

[Ste96] Steinmetz, R., Human Perception of Jitter and Media Synchronization. IEEE Journal on Selected Areas in Communication, Vol. 14, No. 1, January 1996, pp. 61-72.

[Sti99] Stipp, H., Convergence now? The International Journal on Media Management, volume 1, issue 1, autumn/ winter 1999, pp. 10-13.

[Sul98] Sullivan, S. C. et al., Programming with the Java Media Framework. Wiley computer publishing, USA, 1998.

[Sun01a] The J2EE Tutorial. Sun Microsystems, 2001,
http://java.sun.com/j2ee/tutorial/download.html. [3.4.2002]

[Sun01b] J2EE design patterns, Model-View-Controller architecture. Sun Microsystems, blueprints, 2001,
http://java.sun.com/blueprints/patterns/j2ee_patterns/model_view_controller/index.html. [3.4.2002]

[Söd01] Södergård, C. (editor), Integrated news publishing – Technology and user experiences, report of the IMU2 project. VTT publications, Espoo, Finland, 2001.

[TBM99] Tolba, O., Briceño, H., McMillan, L., Pure Java-based streaming MPEG player. Proc. SPIE Vol. 3528, Multimedia Systems and Applications,

January 1999, pp. 216-224. [Also: http://graphics.lcs.mit.edu/cv/mpeg/spie3528-26/spie3528-26.pdf, 3.4.2002]

[Vog00] Vogt, C., The DVB MHP specification – a guided tour. World Broadcasting Engineering, March 2000.